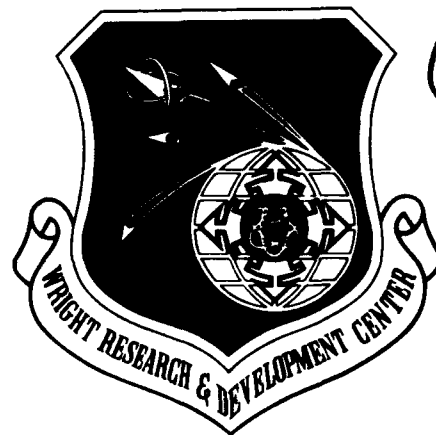




WRDC-TR-90-8007  
Volume III  
Part 7



INTEGRATED INFORMATION SUPPORT SYSTEM (IISS)  
Volume III - Configuration Management  
Part 7 - SCM Administrator's Manual

D. Wagner, M. Foster

Control Data Corporation  
Integration Technology Services  
2970 Presidential Drive  
Fairborn, OH 45324-6209



September 1990

Final Report for Period 1 April 1987 - 31 December 1990

Approved for Public Release; Distribution is Unlimited

MANUFACTURING TECHNOLOGY DIRECTORATE  
WRIGHT RESEARCH AND DEVELOPMENT CENTER  
AIR FORCE SYSTEMS COMMAND  
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6533

**92-12226**

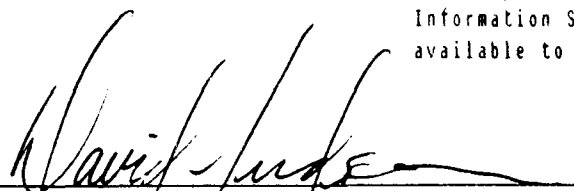


## NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever, regardless whether or not the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data. It should not, therefore, be construed or implied by any person, persons, or organization that the Government is licensing or conveying any rights or permission to manufacture, use, or market any patented invention that may in any way be related thereto.

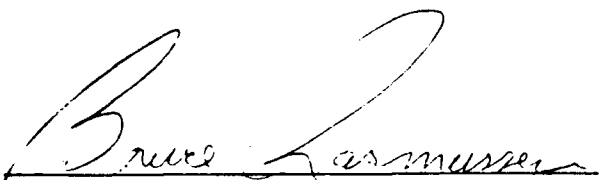
This technical report has been reviewed and is approved for publication.

This report is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations

  
DAVID L. JUDSON, Project Manager  
WRDC/MTI  
Wright-Patterson AFB, OH 45433-6533

25 July 91  
DATE

FOR THE COMMANDER:

  
BRUCE A. RASMUSSEN, Chief  
WRDC/MTI  
Wright-Patterson AFB, OH 45433-6533

25 July 91  
DATE

If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify WRDC/MTI, Wright-Patterson Air Force Base, OH 45433-6533 to help us maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

Unclassified

## SECURITY CLASSIFICATION OF THIS PAGE

## REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified			1b. RESTRICTIVE MARKINGS		
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release; Distribution is Unlimited.		
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE					
4. PERFORMING ORGANIZATION REPORT NUMBER(S) CMA620324000			5. MONITORING ORGANIZATION REPORT NUMBER(S) WRDC-TR-90-8007 Vol. III, Part 7		
6a. NAME OF PERFORMING ORGANIZATION Control Data Corporation; Integration Technology Services		6b. OFFICE SYMBOL (if applicable)	7a. NAME OF MONITORING ORGANIZATION WRDC/MTI		
6c. ADDRESS (City, State, and ZIP Code) 2970 Presidential Drive Fairborn, OH 45324-6209			7b. ADDRESS (City, State, and ZIP Code) WPAFB, OH 45433-6533		
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Wright Research and Development Center, Air Force Systems Command, USAF		8b. OFFICE SYMBOL (if applicable) WRDC/MTI	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUM. F33600-87-C-0464		
8c. ADDRESS (City, State, and ZIP Code) Wright-Patterson AFB, Ohio 45433-6533			10. SOURCE OF FUNDING NOS.		
11. TITLE (Include Security Classification) See Block 19			PROGRAM ELEMENT NO. 78011F	PROJECT NO. 595600	TASK NO. F95600
					WORK UNIT NO. 20950607
12. PERSONAL AUTHOR(S) Control Data Corporation: Wagner, D., Foster, M.					
13a. TYPE OF REPORT Final Report		13b. TIME COVERED 4/1/87-12/31/90	14. DATE OF REPORT (Yr., Mo., Day) 1990 September 30		15. PAGE COUNT 23
16. SUPPLEMENTARY NOTATION WRDC/MTI Project Priority 6203					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify block no.)		
FIELD	GROUP	SUB GR.			
1308	0905				
19. ABSTRACT (Continue on reverse if necessary and identify block number)  This manual provides explicit instructions for creating VAX and IBM releases of IISS. General information for administering Software Configuration Management is also provided.  Block 11 - INTEGRATED INFORMATION SUPPORT SYSTEM (IISS) Vol III - Configuration Management Part 7 - SCM Administrator's Manual					
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED x SAME AS RPT. DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION Unclassified		
22a. NAME OF RESPONSIBLE INDIVIDUAL David L. Judson			22b. TELEPHONE NO. (Include Area Code) (513) 255-7371		22c. OFFICE SYMBOL WRDC/MTI

DD FORM 1473, 83 APR

EDITION OF 1 JAN 73 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

### FOREWORD

This technical report covers work performed under Air Force Contract F33600-87-C-0464, DAPro Project. This contract is sponsored by the Manufacturing Technology Directorate, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. It was administered under the technical direction of Mr. Bruce A. Rasmussen, Branch Chief, Integration Technology Division, Manufacturing Technology Directorate, through Mr. David L. Judson, Project Manager. The Prime Contractor was Integration Technology Services, Software Programs Division, of the Control Data Corporation, Dayton, Ohio, under the direction of Mr. W. A. Osborne. The DAPro Project Manager for Control Data Corporation was Mr. Jimmy P. Maxwell.

The DAPro project was created to continue the development, test, and demonstration of the Integrated Information Support System (IISS). The IISS technology work comprises enhancements to IISS software and the establishment and operation of IISS test bed hardware and communications for developers and users.

The following list names the Control Data Corporation subcontractors and their contributing activities:

<u>SUBCONTRACTOR</u>	<u>ROLE</u>
Control Data Corporation	Responsible for the overall Common Data Model design development and implementation, IISS integration and test, and technology transfer of IISS.
D. Appleton Company	Responsible for providing software information services for the Common Data Model and IDEF1X integration methodology.
ONTEK	Responsible for defining and testing a representative integrated system base in Artificial Intelligence techniques to establish fitness for use.
Simpact Corporation	Responsible for Communication development.

Structural Dynamics  
Research Corporation

Responsible for User Interfaces,  
Virtual Terminal Interface, and Network  
Transaction Manager design,  
development, implementation, and  
support.

Arizona State University

Responsible for test bed operations  
and support.

TABLE OF CONTENTS

		<u>Page</u>
SECTION 1.0	INTRODUCTION .....	1-1
SECTION 2.0	GENERAL DESCRIPTION OF SCM .....	2-1
SECTION 3.0	UPDATING THE SCM DATABASE .....	3-1
3.1	General Description .....	3-1
3.2	Steps to Use UPDCI .....	3-2
3.3	Steps to Use CVTNEW .....	3-2
SECTION 4.0	VAX RELEASE PROCEDURES .....	4-1
4.1	General Description .....	4-1
4.2	Steps For a VAX Release .....	4-2
SECTION 5.0	IBM RELEASE PROCEDURES .....	5-1
5.1	General Description .....	5-1
5.2	Steps for an IBM Release .....	5-1
SECTION 6.0	MANUAL SCM PROCEDURES .....	6-1

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

## SECTION 1

### INTRODUCTION

The SCM administrative functions are a set of automated procedures to be used when appropriate by the SCM administrator. The SCM administrator uses many VAX and IS/Workbench functions manually to accomplish various restructuring tasks such as moving files, creating new directories, deleting libraries, and so forth. The automated procedures are available to handle large tasks such as creating a release.

This document is divided into five more sections:

- Section 2 gives a general description of SCM, including specific information on all SCM directories.
- Section 3 provides information on updating the SCM database file, which contains records on all IISS files.
- Section 4 provides general information and specific directions for all stages of the creation of a VAX release, excluding the directions for testing the release.
- Section 5 provides the above information for an IBM release.
- Section 6 describes the most commonly used manual procedures.

## SECTION 2

### GENERAL DESCRIPTION OF SCM

IISS Software Configuration Management (SCM) has the following functions:

- Storing current source code while preserving the history of changes to it.
- Controlling changes to source code.
- Facilitating releases with automated functions.

The SCM system consists of Source Code Control System (SCCS), some DCL (Digital Command Language) code created by General Electric, and one C program used to interface between the DCL and SCCS. SCCS, a product of Interactive Systems, provides low-level functions for the storage and changing of source code. The SCCS functions are never seen by the general users of SCM; access to the functions is through the relatively user-friendly DCL functions.

The directories used for SCM are [SIISS], [NIISS], [IISS], [IISSIBM], [CMDB], [IISSCM], and [TIISS]. The directory structure in all of these except [IISSCM] and [CMDB] is similar; under the top directory are the subsystems, and one level below that are the subdirectories. The following gives brief descriptions of the functions of these areas:

- [SIISS] is where almost all source code (standard, variable length file format) is stored. The storage of the source code is controlled by SCCS. Each source code file is given a prefix SX and is stored in a form that separately preserves all changes to the original file. Within an SCCS file, changes for a given release have a corresponding SCCS internal number (6 for Release 1.6, 10 for Release 2.0, 15 for Release 2.5, etc.). The files are created with the SCCS "admin" command and changed with the SCCS "delta" command, and any change level of the file can be printed into a file with the SCCS "get" command. Due to the possible duplicate names of host-dependent code, there are separate directories (.VAX and .IBM) under all directories for storing the host-dependent code. These directories are only used in [SIISS]; during releases, the code for the proper host is pulled into the same directory as generic code.
- [NIISS] is where any source code is stored that cannot be put into SCCS (fixed length record format files). At one time it was used to store PLI object files, but is now used only for .FDL (Forms Definition Language), .MSG (message) files, .SCP (script) files, .SAV (save) files, .BAK (Oracle export) files, .FLR (fixed length



record) files, .ADL (Application Definition Language) files, .CTR (EDS test input) files, and .DP (EDS) files.

- [IISS] is the VAX release directory. Prior to a release, most of its directories are cleared out so that the current source can be brought into it from [SIISS] for compilation and linking. [IISS.COM] is where the VAX release procedures are located. [IISS.BUILD] and [IISS.BUILD.EXECTAPE] is the location of the .COM files that are used for installing the IISS software on a VAX.
- [IISS.BUILD] contains the command procedures used for installing an IISS source code tape. [IISS.BUILD.EXECTAPE] contains the command procedures necessary to install an IISS executable tape.
- [IISSIBM] is the IBM release directory. It serves the same function as [IISS] except that it is for IBM releases. [IISSIBM.COM] is where the IBM release procedures are located. [IISSIBM.BUILD] is the location of the JCL files that are used to build the release on the IBM. Some of the JCL files are permanent, but the CL... and the LK... files must be created for each release to provide the JCL needed to compile and link on the IBM.
- [CMDB] is the SCM database directory. It contains CI.DAT, the indexed sequential file containing information on all source code. The accuracy of CI.DAT is very important since it controls which code goes where into releases. [CMDB] contains NEWITEM.DAT, listing all new files that need to go into CI.DAT, and RETURN.DAT, listing all files that were changed since the previous release. It also contains various cross-reference and data files used for the SCM user functions. Directories under CMDB are used to store information used for particular releases. For example, [CMDB.REL018] contains the command files created for release 1.8, and [CMDB.SXREL018] contains SCCS source files that were deleted from SCM around the time of release 1.8. These release-specific directories are stored on disk for a couple of releases and then are moved to tape and archived.
- [IISSCM] is the production directory for the CM user functions, such as NEWITEM and CHECKOUT.
- [TIISS] is the test directory for releases. All executables, command files, data files, message files, and forms files are moved to this directory (mostly to [TIISS.RUNAREA]) for integration testing and debugging of a release. Upon completion of the testing, appropriate code is moved to the production directory [PIISS]. This includes objects, object libraries and executables.

The SCM Administrator is responsible for maintaining all SCM code, including both user functions and administrative functions. The administrator is also responsible for running administrative functions, both for updating the SCM database and creating releases. The database is always updated at the beginning of each release with all new items, but throughout a release new files may be added. At any time file attributes (e.g. host, subdirectory, or link parameters) may be changed, or obsolete files may need to be deleted. When files are deleted or when certain file attributes are changed, the source file under [SIISS] must be moved. This moving is done automatically by the CI.DAT update program (UPDCI.COM), which must be executed while logged onto [SIISS], due to the protections under [SIISS].

The SCM functions provide a relatively user-friendly interface to SCCS code, provide the desired protections and functionalities, and provide much automation of release procedures. Functionalities provided include the ability to use SCM across disks, across different UIC groups, and from directories lacking world access. In order to provide the latter two functions, it was necessary to use the VAX Install utility to permanently install the SCCS interface executable (INTER.EXE) with DETACH and SYSNAM privileges and to permanently install the SCCS executables DELTA.EXE and ADMIN.EXE with SYSPRV privilege.

The SCM user functions are described separately in the SCM User's Manual, CMU620324001. The administrative functions are described in the following sections.

### SECTION 3

#### UPDATING THE SCM DATABASE

##### 3.1 General Description

The most important file in SCM is one that keeps track of all current files that are part of IISS. This database file must be updated when any file is added to the system or deleted from the system or when any file has various information for it changed. The database file is CI.DAT, an indexed sequential file with each record providing attribute fields for a given file. These attribute fields provide the information needed during a release so that the file will be moved to the proper directory, so that the file will be compiled and linked as needed, and so that the object file will be put into the proper library. The CI.DAT file is copied at the end of each release and is provided to all developers as a reference. The alphabetical and sorted versions are Supplement 1 and Supplement 2 to the SCM User's Manual, CMU620324001. The fields in the file are the following:

- file (8 characters, although 7 characters is the maximum allowable filename length)
- extension (3 characters)
- host (1 character)
- subsystem (5 characters)
- subdirectory (10 characters)
- documentation group (10 characters)
- link command file (12 characters)
- linking parameters (40 characters)

The primary key for the CI.DAT file is the file and extension. Therefore these two fields cannot be altered in a given record. If either file or extension need to be changed for a given file, the file must be deleted from SCM and then newitemed with the new name. The alternate key is file, extension, and host. This alternate key is used for accessing a unique record, and duplicate entries are not allowed.

The two functions that are used to update CI.DAT are UPDCI.COM and CVTNEW.COM. UPDCI is used to alter a given file record or to delete it. CVTNEW is used to add new files that have been newitemed to the database.

UPDCI prompts repeatedly for a filename (FILE.EXT) and allows for either updating the file or deleting it. Pressing <RETURN> at the filename prompt exits the program. If the administrator is updating a record, he is prompted for the changes to each field. Pressing <RETURN> at any prompt keeps

that field unchanged. If any fields are changed so that the [SIISS] file must be moved (such as subdirectory or host), that move takes place automatically. If any records are deleted, the [SIISS] file is moved to an obsolete file location under [CMDB], so that the name can be reused if desired. Because of the [SIISS] file moves, it is necessary to be logged onto [SIISS] to run the UPDCI command file.

CVTNEW is called automatically by VSTART.COM at the beginning of a release. It converts the newitems in NEWITEM.DAT that are not designated for future releases into the correct format for being added to CI.DAT and then adds them. The files in NEWITEM.DAT that are designated for future releases remain in the NEWITEM.DAT file. CVTNEW can be invoked separately for adding any additional newitems that are added during a release. It is called with one parameter designating the current release (6 for Release 1.6, 10 for Release 2.0, 15 for Release 2.5, etc.).

### 3.2 Steps to Use UPDCI

1. Log onto [SIISS]
2. Set your default to [IISS.COM]
3. \$ @UPDCI
4. Enter FILE.EXTENSION and HOST at the prompts
5. View the file information
6. Answer whether you want to update the file information (default is NO)
7. If (6) is NO, answer whether you want to delete the file (default is NO)
8. If (6) is YES, type in answers to the fields you want changed. Pressing <RETURN> at any prompt keeps that field unchanged. If you want the field changed to a blank field, type in BLANK.
9. The file will then be updated and you will be prompted for another file and extension. Press <RETURN> when you have finished updating files and wish to exit the program.

### 3.3 Steps to Use CVTNEW

1. Log onto [IISS]
2. Since CVTNEW will delete the most recent version of NEWITEM.DAT, you should save it in the release directory by:

```
$ SET DEFAULT [CMDB.Rel0xx]
$ Edit Newitem.DAT
  * INCLUDE [CMDB]Newitem.DAT
  * EXIT
$ Purge Newitem.DAT
```

where xx is the two digits in the release number (23 for Release 2.3).

3. Set your default to [IISS.COM]

4. \$ CVTNEW y

where y is the SCCS internal number that corresponds with the release number (6 for Release 1.6, 10 for Release 2.0, 15 for Release 2.5, etc.).

5. Check the file [CMDB]CVTNE.ECP. This file lists any exceptions that could not be added to CI.DAT.

## SECTION 4

### VAX RELEASE PROCEDURES

#### 4.1 General Description

A release is a step-by-step sequential procedure that must be checked at each stage to assure that there are no problems. The release account, where the VAX release is done, is IISS. The order that subsystems are released is: (1)IPC, (2)NTM, (3)COMM, (4)UI, and (5)CDM. A VAX release includes the following:

- The CI.DAT file is updated with the addition of all new files (newitems), the deletion of all obsolete files, and the changing of any CI.DAT information as needed.
- The CI.DAT file is sorted by subsystem and subdirectory in order to build command files for the release.
- Command files for the release are built, one subsystem at a time.
- IISS directories are emptied and empty object libraries are created.
- Source code is moved into IISS.
- All compiles, library replaces, and links are done, one subsystem at a time.
- Forms are created.
- The files needed for running IISS are moved to [TIISS], the test directory.
- The testing procedure includes doing precompiles and moving the resultant COBOL files to IISS. These files and the command files created to compile and link them become part of the release.
- When testing is complete, the command files needed to build a release are moved to the [IISS.RELxx.BUILD] (where xx is the release number; e.g., 23 for release 2.3) directory and release tapes are made.

These general steps are normally repeated many times, at least partially, for a release. When a problem is encountered, the steps must either be repeated or the problem area must be manually handled to solve the problem. For instance, if a batch job to compile the User Interface is run and three of the files do not compile correctly, the problem files could be fixed and manually compiled. On the other hand if an include file needed changing and if it were included in twenty or

thirty files, it would probably be easiest to fix the include file and then redo the entire compile batch job. Backtracking is often required during the release. For instance, if a file in the NTM services needs to be modified to solve problems encountered in NTM testing, all links that use the services must be redone. When source code is changed to solve a problem encountered during testing, backtracking must be done as far back as necessary to fix anything affected by the change.

#### 4.2 Steps for a VAX Release

1. Log onto IISS and set your default to [IISS.COM]. This is where the administrative release procedures are located.

2. \$ @VSTART

This procedure can be run only at the beginning of each release. It uses the current release number from [CMDB]RELNUM.DAT to create a release specific directory under [CMDB] for the release, e.g. [CMDB.REL014] for release 1.4. The current NEWITEM.DAT and RETURN.DAT are copied into this directory. CVTNEW.COM is called to update CI.DAT with all newitems for the releases. Other newitems are retained for future releases. When newitems or returns are added in the middle of the release, parts of this procedure must be done manually. That is, NEWITEM.DAT and RETURN.DAT are included into the same files located in the release directory. CVTNEW.COM is called, and RETURN.DAT and NEWITEM.DAT are deleted from [CMDB].

3. If any items in CI.DAT need to be deleted or modified, run UPDCI.COM as described in Chapter 3.

4. \$ @VSUBSYS

This procedure creates separate files in the format of CI.DAT for the subsystems. These files are .TMP files and are put in the release specific directory, [CMDB.REL0xx]. For example, IPC.TMP is a sorted list of the IPC subsystem.

5. Build the command files for all the subsystems:

```
$ @VBLDCOM IPC
$ @VBLDCOM NTM
$ @VBLDCOM COMM
$ @VBLDCOM UI
$ @VBLDCOM CDM
```

All command files to release the subsystems are created. The files created are put into the [CMDB.REL0xx] directory and are keyed to the subsystem specified. For example, for the IPC subsystem, the files created are the following:

- GTIPC.COM will get (retrieve) IPC source code from SIISS and NIISS, putting it in IISS. Include files are copied to the appropriate IISS directories (.SOURCELIB for .INF and .INC files, and .HLIB for .H files).
- ILIPC.COM will do library replaces of .INC files to the include text library, [IISS.SOURCELIB]IISSCLIB.TLB.
- CLIPC.COM will do the compiles.
- OLIPC.COM will do object library replaces and then delete the .OBJ files.
- LKIPC.COM will do the links. All link procedures will place the executable into CMDIR:[RUNAREA]
- All of these command files are built using the IISS system logicals developed for Release 2.3. These logicals must be defined at the group level within any account that can be used to build or run a release. Please refer to the System Administrator's Manual, Pub. No. SUM 620320000 for additional information on this activity.

6. \$ @VCREGET

This creates a file, DOGET.COM, that will retrieve the source code for all subsystems.

7. Now, create the files that will do compiles, library replaces, and links for the subsystems:

```
$ @VCREDO IPC
$ @VCREDO NTM
$ @VCREDO COMM
$ @VCREDO UI
$ @VCREDO CDM
```

One file is created for each subsystem and is put into the [CMDB.REL0xx] directory. The file name is keyed to the subsystem specified. For example, DOIPC.COM is created for the IPC subsystem.

8. \$ @VDELALL

This procedure calls VDELETE for all subsystems to clean out all files from the IISS directories. It also deletes the forms library, include libraries, and all object libraries. All libraries are recreated empty.

9. Set your default to [CMDB.REL0xx] and start the batch job:

```
$ submit/notify/keep/noprinter/queue=EVE_BATCH DOGET.COM
```



This batch job will pull all source code for all subsystems into IISS and copy include files to the include libraries. This will take hours to complete. When it is finished a batch output log will be produced. Check the log for any errors, and correct them before proceeding.

10. Now the various subsystems can be built. Running VDORUN creates a batch job that does compiles, library replaces, and links for a given subsystem. The batch jobs will take various lengths of time depending upon the size of the subsystem. These runs must be done in the following order because of links that depend on compilations of other subsystems:

```
$ SET DEF [IISS.COM]
$ @VDORUN IPC
$ @VDORUN NTM
$ @VDORUN COMM
$ @VDORUN UI
$ @VDORUN CDM
```

A batch output log will be produced at the end of each batch job. You also may want to "comment out" the links execution command until all compiles have been completed successfully. Check it for errors. Some links in the UI and the CDM will not work at this time because they depend upon precompiled modules. These will be done manually at a later time.

11. The forms for the User Interface must now be created by running the stand-alone version of FLAN to create all the .FD files from the .FDL files. The .FDL files are located in [IISS.FD], and that is where the forms will be created. Set your default to [IISS.FD] and run the stand alone flan executable using the following command:

```
$ Run CMDIR:[RUNAREA]FLANSA
```

Prior to running FLANSA, you need to reassign the IISSULIB logical using the following command:

```
$ assign [IISS.FD] IISSULIB
```

Make a list of all .FDL files in the directory. Run FLANSA once for each .FDL file. When you run FLANSA, at the file: prompt enter the name of the .FDL file, and all .FD files defined in that file will be created.

12. Although the release basically is built, precompiling is required to build the NDDL and RAP executables. Testing can actually be started after the NTM has been built, and other subsystems can be tested as they are built.) The testing is done in the TIISS account, under [TIISS.RUNAREA]. After the directories in TIISS have been cleaned out, log onto TIISS, and copy

(rename if disk space is a problem) all files required for testing. These files include executables, test input, test compare, etc.

13. Directions for integration testing are found in other manuals. (See IISS Documentation Description, CMB620300000, for a list of the documents.) Normally the IPCs and COMM are tested first. Then the NTM tests are run. Then the UI tests that are not dependent upon a precompiled module are run. Prior to and during the running of these tests, NDDL must be built using the NDML precompiler while you are logged on the IISS account. Documentation for this activity is provided for each release by subcontractors responsible for the subsystems requiring precompilation. Most information on this is contained in an attachment to the CDMP Test Case Report Documentation, called Define/Construct NDDL, Pub. No. UM 620341401.

Once the NDDL executable has been built successfully, move the executable to the TISS account using the copy or rename command. The CDM Tools and Reports will be built using a similar method and the documents provided by the relevant subcontractor. Again, refer to the IISS Documentation Description, Pub. No. CMB620300000, for a list of relevant build documents.

14. When testing is deemed complete and all required changes have been made to SCM, IISS, and [CMDB.REL0xx], IISS can be prepared for making the release tape. Directories, such as [IISS.NTM...] and [IISS.UI...] should be purged, and .LIS files and .LOG files should be deleted.
15. The .BUILD and .BUILD.EXEC directories must be made current. The following procedures are done manually:
  - Check all command files in [IISS.BUILD], such as LNKSOUR.COM and [IISS.BUILD.EXEC]LNKIISS.COM to see if they are current and correct.
  - Now purge the files in [IISS.BUILD] and [IISS.BUID.EXECTAPE].
16. The VAX release tapes can now be made. As of Release 2.3, both an EXECUTABLE and SOURCE Release tape are required. First, you must update all release specific command procedures used during the installations. These procedures reside in [IISS.BUILD] for the source release tapes and [IISS.BUILD.EXEC] for the executable release tape. Next, there are command procedures in [IISS.TAPE] for establishing the release environment and cutting the tapes.

\$ SET DEFAULT [IISS.TAPE]

\$ @RELDIR x y

- where x is the first number (example, 2 for 2.3) of the current release number and y is the second number (example, 3 for 2.3).
- NOTE: This procedure creates a directory structure in the form of [IISS.RELXY ...] on top of the normal subsystem directory structure. If new subsystems or sub-directories have been added you must update this command procedure, as well as the other procedures used in these steps.

17. The various modules needed for a release tape must be moved to the release directory structure using two different command procedures. Execute the following command:

- \$ @RELREN x y

- where x is the first number (example, 2 for 2.3) of the current release number and y is the second number (example, 3 for 2.3). This procedure renames all command procedures, object libraries, executables, and data files needed to LINK and RUN IISS. This does not rename any SOURCE code.

18. The executable release tapes should now be created. You should create/cut any and all required executable tapes at this time. You can create these tapes and the source code tapes using the following command procedure.

\$ @CUTTAPE

- This procedure is used for cutting release tapes for the current release only. It will tell you the current release number and it will prompt you for the following tape drive:

"ENTER DESTINATION TAPE:"

- After having the operator mount a tape and obtaining the name of the tape drive, enter the tape name.
- Next, the procedure prompts you for whether or not you are a SOURCE CODE release tape.

"Are you creating a SOURCE code Release tape (Y/N):"

- Enter 'Y' or 'N', in this case it should be 'N', since you are creating all executable tapes needed at this time.

- The procedure now reminds you to update the Install command procedures and gives you the opportunity to exit the tape cutting procedure if necessary.
- Finally this procedure prompts you for the density at which the tape should be cut.

"At what density should the tape be cut (1600 or 6250):"

Note: If you are cutting a release tape to fullfill a government contract requirement if must be cut at 1600 bpi.

- Enter the density required for the tape (be sure and verify this with the organization for whom you are creating the tape). If you enter 1600, you will be creating a multivolume tape.
  - The procedure then will proceed with creating the tape. If you receive any unusual errors, unrecoverable media errors, etc., you must restart this procedure. Refer to the VAX/VMS backup utility reference manual, if necessary.
  - Finally, the procedure will notify you when the tape is complete. This procedure dismounts the tape, but does not unload it, this allows you to manually have it backed up and listed (BACKUP/LIST) and to verify the tapes contents. Again, refer to the VAX/VMS backup reference manual for more information if required.
19. After you have created the required executable tapes, begin the procedure to create the required source code tapes. First, move/rename all SOURCE code modules to the release directory structure described in step 16. Then execute the following:

\$ @SRELREN x y

- where x is the first number (example, 2 for 2.3) of the current release number and y is the second number (example, 3 for 2.3). This will move all the source code modules to the release directory environment.
20. You may now execute the same procedure used to cut the EXECUTABLE tapes when creating the SOURCE code tapes. (Refer to Step 18.)

Note: The only thing different about creating the source code tapes using CUTTAPE.COM is the answer you provided the second prompt. You should now enter 'Y' indicating that you are creating a SOURCE CODE tape. (This executes a different backup statement which excludes executables).

21. After you have completed the creation of all release tapes required at this time, you must now move/rename all of the IISS modules back to the IISS integration environment. First, you must move/rename all of the SOURCE CODE modules back to this environment by executing the following command:

\$ @SRELDEL x y

- where x is the first number (example, 2 for 2.3) of the current release number and y is the second number (example, 3 for 2.3).
- Note: This procedure returns the release directory environment to its executable state and you would cut more EXECUTABLE Release tapes, if required.

22. Now, you must move/rename the EXECUTABLE modules to the IISS integration environment using the following command:

\$ @RELDEL x y

- where x is the first number (example, 2 for 2.3) of the current release number and y is the second number (example, 3 for 2.3).
- This will move/rename all EXECUTABLE modules to the IISS integrate environment and deletes the release directory environment used for cutting release tapes.

Note: If more tapes are required at a later date you must start at step 16 and proceed step by step through step 22.

- Finally, these steps are dependent upon the other; therefore, you cannot skip certain steps, i.e., do not execute SRELREN.COM to create a source code tape without first executing RELREN.COM.

23. The VAX Installation Guide for Executable Code (OM 620324001, 31 May 1988) and the VAX Installation Guide for Source Code (OM 620324003, 31 May 1988) should be updated to be provided with the release tape.

24. It is recommended that a trial build be done with the tape on a VAX with the appropriate hardware and software. If any problems are encountered in the build, the appropriate corrections may be in [IISS] prior to the making of another release tape. Ideally, this procedure is repeated as tape. Ideally, this procedure is repeated as necessary until a tested and problem-free final version of the tape is produced.

25. Once the release is completed, the release number should be updated in RELNUM.DAT by entering the following command (with your default set to [IISS.COM]):

\$ @VEND rein

where rein is the next release number (e.g., 2.1)

## SECTION 5

### IBM RELEASE PROCEDURES

#### 5.1 General Description

IBM release tapes are created from an IBM release account on the VAX, IISSIBM. The directory structure of IISSIBM is similar to that of IISS, the VAX release directory. Code that is used for building the release on an IBM is located in [IISSIBM.BUILD], and release procedures that are used for creating the release tape from the VAX are located in [IISSIBM.COM].

The IBM release includes the following:

- VAX release procedures are used as needed to update CI.DAT and create the .TMP files that are the sorted lists for each subsystem.
- The IBM release directory is cleaned up in preparation for the new release.
- Command files for the release are built, one subsystem at a time.
- Source code is moved into IISSIBM.
- JCL files to do compiles and links on the IBM are moved to the [.BUILD] directory.
- A directory listing of all files to go on the tape is created.
- The release tape is created.
- The contents of the tape must be loaded onto an IBM to test the build procedure and to test the IBM version of IISS.

#### 5.2 Steps for an IBM Release

1. Do steps 1,2,3, and 4 of the VAX release procedures (Section 4.2) as needed, unless they have already been done for the VAX release.
2. Log onto IISSIBM and set your default to [IISSIBM.COM]. To do the necessary prerelease cleanup, run the following:

```
$ @IDELALL
```

3. Build the command files to release the subsystems:

```
$ @IBLDCOM IPC  
$ @IBLDCOM NTM  
$ @IBLDCOM COMM  
$ @IBLDCOM UI  
$ @IBLDCOM CDM  
$ @IBLDCOM CICS
```

4. Pull the source code into IISSIBM:

```
$ SUBMIT/NOTIFY GTIPCxx  
$ SUBMIT/NOTIFY GTNTMxx  
$ SUBMIT/NOTIFY GTCOMMxx  
$ SUBMIT/NOTIFY GTUIxx  
$ SUBMIT/NOTIFY GTCDMxx  
$ SUBMIT/NOTIFY GTCICSxx
```

where xx is the release number (23 for Release 2.3). Check the logs that are automatically printed at the end of the batch jobs. These batch jobs may be run concurrently.

5. Move all the compile and link files to the [.BUILD] directory:

```
$ copy CL*.JCL [IISSIBM.BUILD]*  
$ copy LK*.JCL [IISSIBM.BUILD]*
```

6. While your default is still set to [IISSIBM.COM], it is necessary to create the log datasets:

```
$ @IBLDLOG
```

7. The tape can now be created. Log onto IISSIBM on the hard copy terminal and set your default to [IISSIBM.COM]. Mount a tape and then start creating it:

```
$ @CRTTAPE
```

When the tape is finished, print the listing:

```
$ PRINT TAPE.MST
```

8. The Installation Guide (OM 620324002) should be updated to be provided with the release tape. Building and testing must be done on the IBM, and any needed corrections must be made on the VAX prior to making a new tape.



## SECTION 6

### MANUAL SCM PROCEDURES

If everything in a release were to work perfectly the first time, it would only be necessary to use the automated procedures. However this is rarely the case, so that knowing how to do parts of the procedures manually is a necessity.

When a file needs to be changed for any reason after the release is started, the new version can be pulled into IISS. The file must be changed in Configuration Management by using the user functions CHECKOUT and RETURN. This is normally the responsibility of the developer in charge of the particular subsystem. Then you must log onto IISS and set your default to the directory where the changed file resides. Delete the old file. Then bring in the new version with the following SCCS command:

```
$ GET -Rn IISS_CM~/SIISS/subsys/subdir/SXfilename.extension
```

Fill in the appropriate values for subsystem, subdirectory, filename, extension, and n (the SCCS internal number, e.g. 10 for Release 2.0, 15 for Release 2.5, or 29 for Release 3.9).

When a new file has been pulled in, all needed compiles, library replaces, and links must be performed to accommodate the change into the release. These are accomplished with standard VMS procedures. Refer to the appropriate Product Specification(s) for information on what will be affected by the changed module or include file. The format of some commonly used procedures is as follows:

```
$ COBOL/ANSI/NOLIST filename
```

```
$ FORTRAN/NOLIST filename
```

```
$ CCX filename
```

```
$ LIBRARY/REPLACE xOLB.OLB filename.OBJ
```

where x is the first 4 characters of the directory.

```
$ COPY filename.H [IISS.HLIB]*
```

```
$ COPY filename.INC [IISS.SOURCELIB]*
```

```
$ LIBRARY/REPLACE/TEXT IISSCLIB filename.INC
```

```
$ COPY filename.INF [IISS.SOURCELIB]
```

Deletes and purges should be done throughout as needed to keep IISS current and clean.